

The ascending sort feature is similar to the sort function of python `sort(a,key=lambda a:key(a))`

Here we can sort the array according to the key feature.

If nothing is given in key it sorts normally

Note: When the key value is same elements are sorted according to the inplace sorting

Examples

Sort Array in Ascending and Descending order

Array :

key :

Copy Text

Copy Text

According to the primes

2 3 5 and 7 are primes so in ascending order they are at last and in descending order they are at first

Sort Array in Ascending and Descending order

Array :

key :

Copy Text

Copy Text

Sorting according to the sum of the array.

Sum of array [1,10] is maximum i.e. 11 so it is at last and sum of array [2,3] is minimum i.e. $2+3=5$ so it is at first in ascending order.

Sort Array in Ascending and Descending order

Array :

1 2 3 4 5 6 7 8 9 10 11 12 13

key :

__builtin_popcountll

Sort Array

Clear Array

Back

Copy Text

1 2 4 8 3 5 6 9 10 12 7 11 13

Copy Text

7 11 13 3 5 6 9 10 12 1 2 4 8

Sorting according to the number of ones in the binary representation of the number.

1 2 4 8 binary are "1" "10" "100" "1000" so the number of ones are minimum so they are at first

Sort Array in Ascending and Descending order

Array :

[1,10] [2,9] [4,8]

key :

function a(arr){return arr[1]};a

Sort Array

Clear Array

Back

Copy Text

4,8 2,9 1,10

Copy Text

1,10 2,9 4,8

Sorting according to the second value of the array.

You can also customize the function as seen here. Simply make a function (javascript) and write its name separated by a semicolon.

[Explore the docs »](#)

Sort Array in Ascending and Descending order

Array :

key :

Copy Text

Copy Text

Sorting according to the `a%2` custom function.

[Explore the docs »](#)

Sort Array in Ascending and Descending order

Array :

key :

Copy Text

Copy Text

Sorting according to the length.

The strings should be entered in `" "` or `' '` i.e. double quotes or single quotes.

The length of the string 19 and 99 are 2 so they are at last

8 comes before 1 because they have the same length and so they occur according to the index in input i.e. inplace sorting.

[Explore the docs »](#)

Sort Array in Ascending and Descending order

Array :
`range(100)`

key :
`function r(){return random(10000)};r`

[Sort Array](#) [Clear Array](#) [Back](#)

Copy Text

```
19 60 18 61 40 66 14 26 95 0 1 2 16 84 77 80 7 51 39 22 74 96 11 6 90 72 93 83 57 37 29 67 32 50 10 12 64 31 27 44 49 15 45 75 3 25 86 47 46 70 23 42 20 79 17
63 76 36 21 38 81 52 92 9 13 99 87 82 59 43 98 91 33 56 5 34 68 97 88 4 62 41 85 58 69 28 65 53 35 48 30 8 89 94 71 78 54 55 73 24
```

Copy Text

```
59 69 8 79 87 23 63 92 35 68 99 6 46 29 30 86 27 13 56 60 90 4 10 48 37 58 98 82 72 28 61 31 84 45 0 53 44 67 51 22 3 96 33 54 5 40 77 52 93 66 75 36 83 38 73
41 74 91 78 71 11 95 16 42 43 70 49 25 64 97 26 19 34 20 62 88 57 55 85 17 94 39 80 32 1 15 89 14 21 7 65 18 50 2 12 76 24 47 9 81
```

Shuffling the array randomly

In the key you can provide the following

1. `bit_length`
2. `len`
3. `isPrime`
4. `__builtin_popcountll`
5. `__builtin_parityll`
6. `__builtin_clzll`
7. `__builtin_ctzll`
8. `is_sorted`
9. `sum`
10. `mex`

And many more...

Other key function can be made using custom javascript functions.